Homework - 8

Hariharan Seshadri

School of ECE, Purdue University

hseshadr@purdue.edu

This report has the following segments:

1. Introduction

The goal of this exercise is given a set of images of the calibration (3 or more images from different angles) , the camera calibration matrix (intrinsic parameters) and the pose of the camera with respect to the calibration pattern (extrinsic parameters) is to be estimated.

2. Corner Extraction

Corner Extraction requires multiple steps and a post-processing stage.

Canny edge detector is used as a pre-processing step to the Hough Transform. These first find the strong edges (lines) and any disjoint edges found on the same line will be found in voting in the parameter space (Hough Space). The two thresholds I used for Canny edge detector were 255*1.5 and 255.

The next step is to use the output of Canny Edge Detector to find lines in the parameter space. We know that any line in the image plane has a unique ρ and θ where ρ is the shortest (perpendicular) distance from the origin to the line and θ is the angle made by the perpendicular with the horizontal positive axis.

Any pixel (x,y) may line on a line given by:

ρ = x cosθ + y sin θ

Each pixel will give rise to a sinusoid in the ρ -θ space for positive ρ values andθ varying from 0 to 2Π

Thus, by "voting" (i.e.) thresholding for number of intersections in the ρ -θ space, the lines that exist in the image plane may be identified. Recall that the point of intersection in the ρ -θ space actually corresponds to a unique line and the number of intersections at that point of intersection in the ρ -θ space is the number of pixels that lie on the line.

I used the Probabilistic Hough transform in OpenCV and the function parameters I used were:

1. rho: Distance Resolution in pixels. I used 1 pixel

2. theta: Angle resolution in radians. I used $\Pi/180$ radians

3. thresold: Minimum number of pixels required on a line (i.e.) number of intersections in the ρ -θ space. I used 100 pixels

4. minLineLength: Minimum line length. I used 100 pixels

5. maxLineGap: Maximum allowed gap between points on the same line to link them. I used 600 pixels

An issue I faced was that multiple lines were picked up for the same line in the image plane. So, I had to use the algorithm in *Algorithm 1* to pick the right corners.

Algorithm 1 Identifying correct corners and labeling them

Step 1: Initialize *corner_points* and *good_lines* to be empty data structures

Step 2: For a given pair of lines, identify the angle between them. If they are less than *decision_threshold_1* or greater than *180 - decision_threshold_1,* ignore the pair (I used *decision_threshold_1* = 30 degrees)

Step 3: If the angle is greater than *decision_threshold_1* and less than *180 - decision_threshold_1*, identify the corner given by them as the point of intersection of the pair of lines. If there already exists a corner within a distance of *decision_threshold_2* (I used 15 pixels), ignore the generated corner. Else, add the corner to *corner_points.*

Step 4: If this pair of lines generates a good corner (i.e.) steps 2 and 3 are favourable , then add this pair of lines to *good_lines*

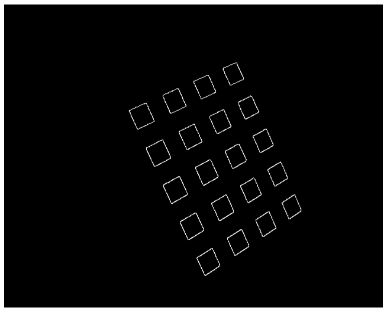
Step 5: Repeat from step 2 until all pairs of lines are exhausted

Step 6: Once all the lines have been identified, bin them into horizontal and vertical lines based on their slopes

Step 7: Sort all horizontal lines based on Y-intercepts and all vertical lines based on X-intercepts

Step 8: Use the sorted horizontal and vertical lines (based on intercepts) to label the corners (given by the intersection of a horizontal line and a vertical line) in a certain way

The results for Canny edge detector and Hough Transform (post-processed using *Algorithm 1*) are shown for 3 images.


(a) Canny Detector

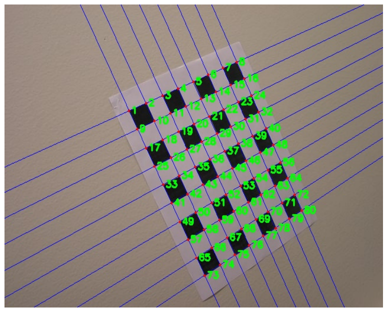
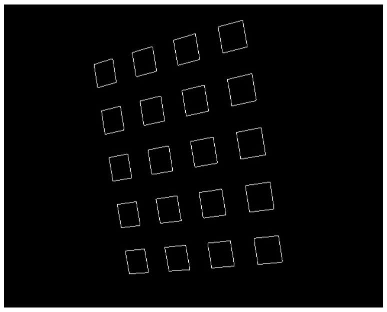
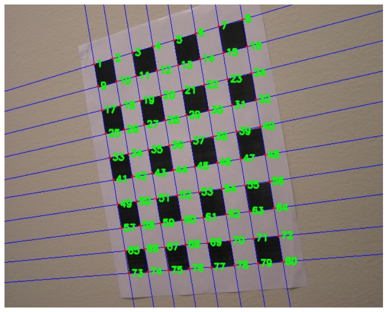

(b) Hough Transform Results

Figure 1: Canny detector and Hough Transform

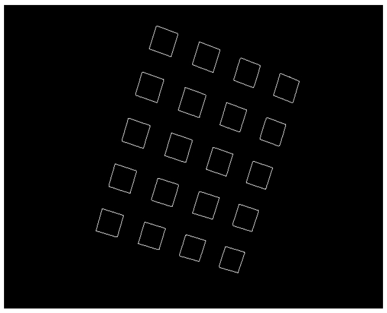


(a) Canny Detector

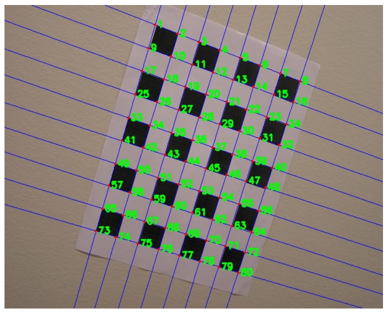


(b) Hough Transform Results

Figure 2: Canny detector and Hough Transform



(a) Canny Detector



(b) Hough Transform Results

Figure 3: Canny detector and Hough Transform

3. Homography Estimation (Recall)

After identifying the corner points, the homography between the world plane (real world coordinates) and the image plane (i.e.)

$x_{\text{image}} = H\ x_{\text{world}}$

The matrix form of the equations given by correspondences is given by:

$A.\vec{h} = B$

where, $\vec{h}$ contains the 8 unknowns - (h_{11} to h_{32}) . We explicitly set $h_{33}= 1$ in this case
and A and B are defined as shown below

Since, each correspondence generates two equations, the two rows in the A and B matrices for one correspondence will be given as:

$$\begin{bmatrix} x_{world} & y_{world} & 1 & 0 & 0 & 0 & -x_{world}x_{image} & -y_{world}x_{image} \\ 0 & 0 & 0 & x_{world} & y_{world} & 1 & -x_{world}y_{image} & -y_{world}y_{image} \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{bmatrix} = \begin{bmatrix} x_{image} \\ y_{image} \\ . \\ . \\ . \\ . \\ . \\ . \end{bmatrix}$$

Thus, we need at least 4 correspondence (here, I used all 80) to compute homography which is given by:

$\vec{h} = (A^tA)^{-1}A^tB$

Thus, the homography between the world plane and EACH IMAGE PLANE is esimated.

4. Intrinsic Camera Parameters Estimation

The following result will be used in estimating intrinsic parameters of the camera-

Image of absolute conic, $W = K^{-T}K^{-1}$

Hence, the goal is to estimate the elements of W. Since, any real world plane samples the absolute conic at two points - the plane's circular points (1 i 0) and (1 -i 0), and if $[\vec{h}_1\ \vec{h}_2\ \vec{h}_3]$ is the homography that relates a real world plane to the image plane, the points transform as $(\vec{h}_1 + i\ \vec{h}_2)$ and $(\vec{h}_1 - i\ \vec{h}_2)$. These points lie on the image of the absolute conic W.

Hence,

$(\vec{h}_1 + i\ \vec{h}_2)^T W\ (\vec{h}_1 + i\ \vec{h}_2) = 0$ and $(\vec{h}_1 - i\ \vec{h}_2)^T\ W\ (\vec{h}_1 - i\ \vec{h}_2) = 0$.

This gives us two equations -

1) $\vec{h}_1^T\ W\ \vec{h}_1 - \vec{h}_2^T\ W\ \vec{h}_2 = 0$

2) $\vec{h}_1^T\ W\ \vec{h}_2 = 0$

Since, W is a 3X3 matrix which is symmetric and homogeneous (hence 5 Degrees of Freedom), we need a minimum of 5 equations to estimate its values. Or equivalently, we need atleast 3 images (hence atleast 3 world plane to image plane homographies) of the calibration pattern on a plane to estimate W. Once, W is estimated, we can use Cholesky decomposition to estimate K.

5. Extrinsic Camera Parameters Estimation

From the basic relation between the plane-plane homography H, Intrinsic parameters K , Rotation Matrix R and translation t :

K$[\overrightarrow{r}_1 \overrightarrow{r}_2 \overrightarrow{t}\ ] = [\ \overrightarrow{h}_1 h_2 \overrightarrow{h}_3]$

We hence get,

$\overrightarrow{r}_1 = \mathrm{K}^{-1}\overrightarrow{h}_1$

$\overrightarrow{r}_2 = \mathrm{K}^{-1}\overrightarrow{h}_2$

$\overrightarrow{r}_3 = \overrightarrow{r}_1 \mathrm{X}\ \overrightarrow{r}_2$

$\overrightarrow{t} = \mathrm{K}^{-1}\overrightarrow{h}_3$

Since, the rotation matrix is an orthonormal matrix, the individual columns must be scaled accordingly. The scale factor is given by: $\xi = \frac{1}{||K^{-1}\overrightarrow{h1}||}$. Hecne, the new equations become:

$\overrightarrow{r}_1 = \xi \mathrm{K}^{-1}\overrightarrow{h}_1$

$\overrightarrow{r}_2 = \xi \mathrm{K}^{-1}\overrightarrow{h}_2$

$\overrightarrow{r}_3 = \overrightarrow{r}_1 \mathrm{X}\ \overrightarrow{r}_2$

$\overrightarrow{t} = \xi \mathrm{K}^{-1}\overrightarrow{h}_3$

We also need to make sure that the condition $\mathrm{R}^{\mathrm{T}}\mathrm{R} = \mathrm{I}$ is met, hence we need to “condition” the rotation matrix. This is done by computing the SVD of the R matrix:

$\mathrm{R} = \mathrm{UVD}^{\mathrm{T}}$and setting the singular values to 1, hence $\mathrm{R}_{\mathrm{conditioned}} = \mathrm{UV}^{\mathrm{T}}$

6. Non-Linear Least Squares Optimization

There is scope for further refinement in the intrinsic and extrinsic parameters based on true location of a pixel in the image plane and the “estimated projection” of the real world point onto the image plane using K,R and t.

The cost function to be minimized is:

$d^2_{geom} = \sum_{\mathrm{i}}\sum_{\mathrm{j}} ||\ \overrightarrow{x}_{\mathrm{i,j}} - \mathrm{K}[\mathrm{R}_{\mathrm{i}}|\mathrm{t}_{\mathrm{i}}]\ \overrightarrow{x}_{\mathrm{M,i,j}}||$

where, $\overrightarrow{x}_{\mathrm{i,j}}$ is the j$^{\mathrm{th}}$salient point in the i$^{\mathrm{th}}$image , and

$\overrightarrow{x}_{\mathrm{M,i,j}}$is the corresponding co-ordinate in the model or calibration pattern.

This can be re-written as:

$d^2_{geom} = \sum_{\mathrm{i}}\sum_{\mathrm{j}} ||\ \overrightarrow{x}_{\mathrm{i,j}} - \mathrm{f}(\mathrm{K},\ \overrightarrow{r}_{\mathrm{i}}\ ,\ \overrightarrow{t}_{\mathrm{i}},\ \overrightarrow{x}_{\mathrm{M,i,j}})\ ||$

where, $\overrightarrow{r}_{\mathrm{i}}$ is the Rodrigues representation of the Rotation Matrix R_{i}. This is important because R has only 3 degrees of freedom and the Rodrigues vector contains a vector consisting of three elements which is the unit vector along the axis of rotation and whose magnitude is the angle of rotation.

7. Reprojection onto "Fixed Image" and error computation

One of the images was kept as the fixed image. For the given dataset, it was image 11 and for my dataset it was image 1. To reproject a point from an image into the fixed image, I first estimated the homography for a given image to the world plane and then the homogrpaphy from world plane to the fixed image plane.

$\vec{x}_{\text{fixed}} = (H_{\text{world\_to\_fixed}})\ (H_{\text{world\_to\_image}})^{-1}\ (\vec{x}_{\text{image}})$, where $\vec{x}_{\text{fixed}}$ and $\vec{x}_{\text{image}}$ are homogeneous coordinate representations.

Once reprojection was done, euclidean error was measured for all 80 points of the image and an average for one image was obtained.

This was done BEFORE and AFTER non-linear optimization. In order to construct the homographies after non-linear optimization of K, R, t is used the relation:

$H^{\text{refined}} = K^{\text{refined}}[\vec{r}_1^{\ \text{refined}}\ \vec{r}_2^{\ \text{refined}}\ \vec{t}^{\ \text{refined}}\ ]$

Once the refined homographies were estimated, reprojection was tried again and the error was measured.

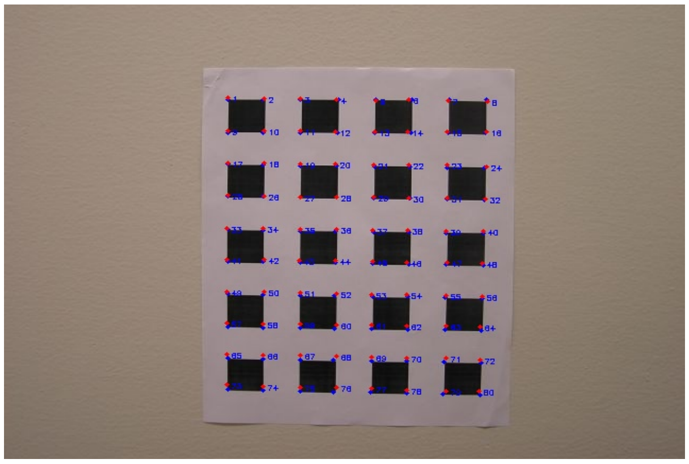
8. Results

(i) Provided Dataset

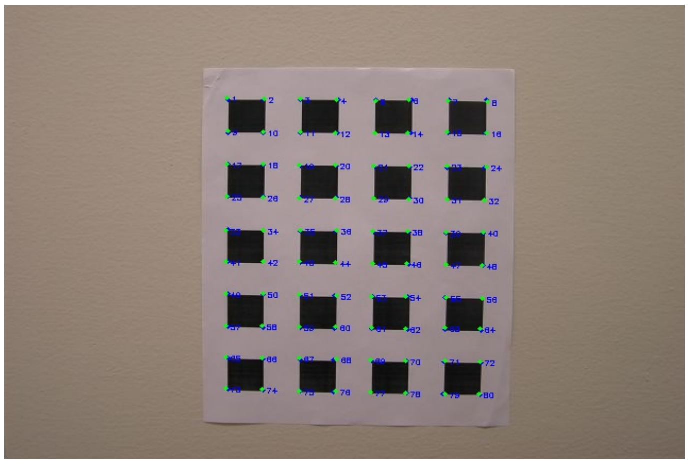
$$K = \begin{bmatrix} 719.86 & 1.66 & 320.23 \\ 0 & 720.06 & 235.8 \\ 0 & 0 & 1 \end{bmatrix}$$

$$K_{\text{refined}} = \begin{bmatrix} 720.05 & 1.90 & 321.73 \\ 0 & 719.79 & 238.51 \\ 0 & 0 & 1 \end{bmatrix}$$

For the following images the effect of non-linear optimization has been depicted. Blue corners indicate the true location of the corner in the "fixed image", red corners indicate the reprojected corner BEFORE optimization and green corners indicate the reprojected corner AFTER optimization. We can visualize the improvement as the green corners are closer to the corresponding blue corners than the red corners.



(a) Before optimization



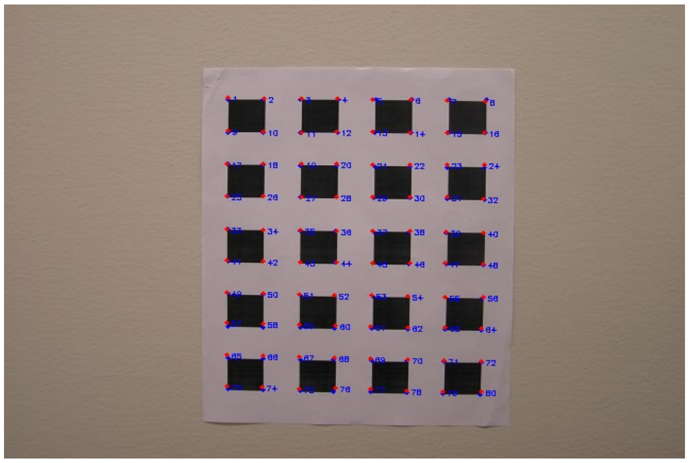
(b) After optimization

Figure 4: Visual verification of optimization (Reprojection of image 3 points)

The extrinsic parameters for this pose of the camera (Image 3 Dataset 1) are:

$$[\mathrm{R}|\mathrm{t}] = \begin{bmatrix} 0.87 & 0.16 & -0.45 & -6.15 \\ -0.197 & 0.97 & -0.027 & -8.16 \\ 0.43 & 0.13 & 0.89 & 48.9 \end{bmatrix}$$

$$[\mathrm{R}_{\text{refined}}|\mathrm{t}_{\text{refined}}] = \begin{bmatrix} 0.88 & 0.16 & -0.43 & -6.3 \\ -0.19 & 0.97 & -0.03 & -8.4 \\ 0.42 & 0.11 & 0.89 & 49.08 \end{bmatrix}$$


(a) Before optimization

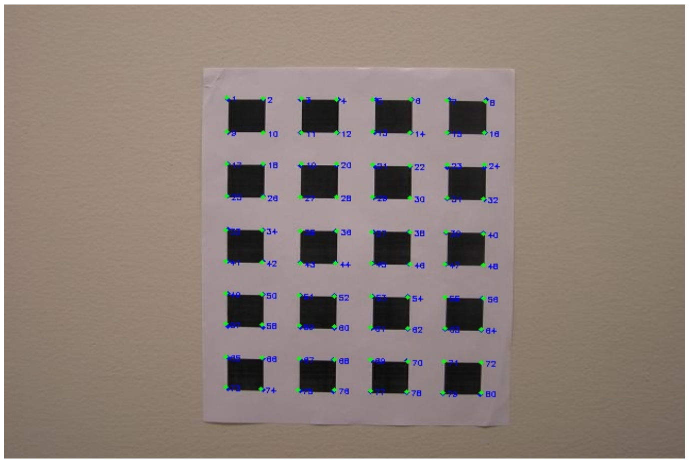
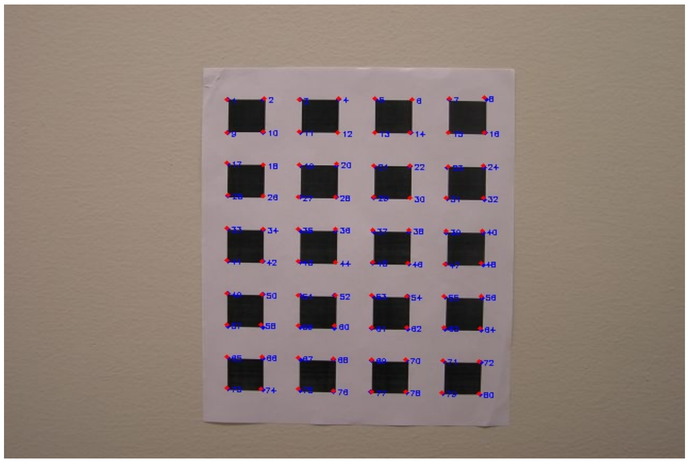

(b) After optimization

Figure 5: Visual verification of optimization (Reprojection of image 6 points)

The extrinsic parameters for this pose of the camera (Image 6 Dataset 1) are:

$$[\mathrm{R}|\mathrm{t}] = \begin{bmatrix} 0.98 & 0.13 & -0.07 & -8.5 \\ -0.13 & 0.98 & -0.03 & -8.15 \\ 0.06 & 0.04 & 0.99 & 61.9 \end{bmatrix}$$

$$[\mathrm{R}_{\mathrm{refined}}|\mathrm{t}_{\mathrm{refined}}] = \begin{bmatrix} 0.98 & 0.13 & -0.069 & -8.66 \\ -0.13 & 0.99 & -0.035 & -8.4 \\ 0.06 & 0.04 & 0.99 & 62.06 \end{bmatrix}$$


(a) Before optimization

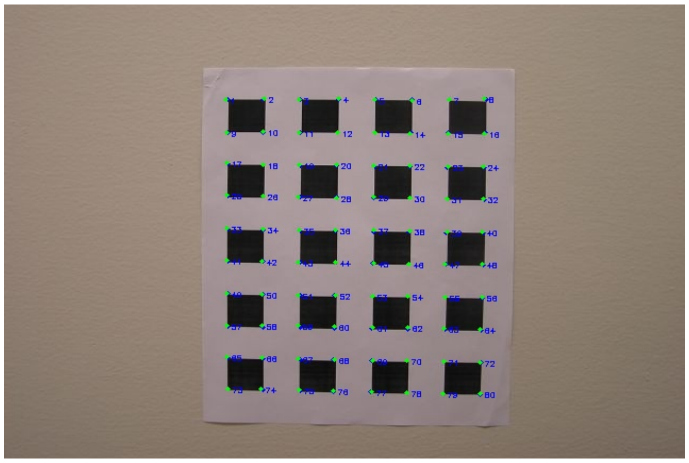
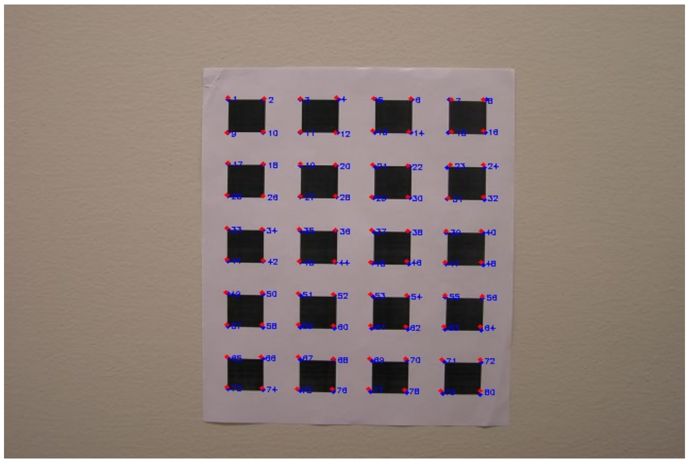

(b) After optimization

Figure 6: Visual verification of optimization (Reprojection of image 9 points)

The extrinsic parameters for this pose of the camera (Image 9 Dataset 1) are:

$$[\mathrm{R}|\mathrm{t}] = \begin{bmatrix} 0.88 & -0.089 & 0.44 & -5.19 \\ -0.17 & 0.84 & 0.5 & -9.01 \\ -0.42 & -0.52 & 0.73 & 58.24 \end{bmatrix}$$

$$[\mathrm{R}_{\mathrm{refined}}|\mathrm{t}_{\mathrm{refined}}] = \begin{bmatrix} 0.88 & -0.09 & 0.45 & -5.2 \\ -0.17 & 0.84 & 0.50 & -9.21 \\ -0.42 & -0.52 & 0.73 & 58.13 \end{bmatrix}$$


(a) Before optimization

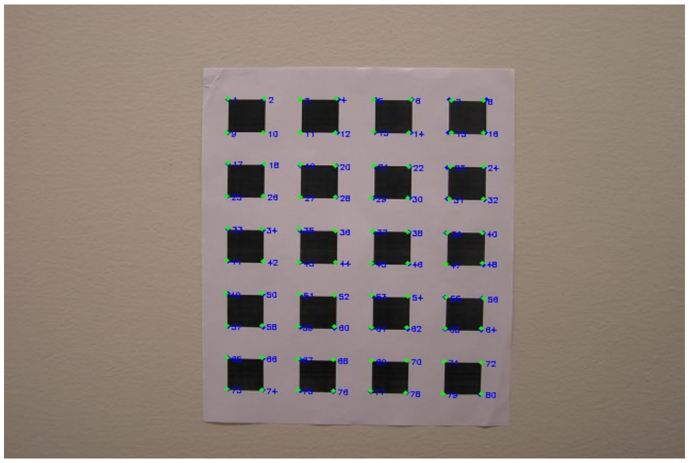

(b) After optimization

Figure 7: Visual verification of optimization (Reprojection of image 12 points)

The extrinsic parameters for this pose of the camera (Image 12 Dataset 1) are:

$$[\mathrm{R}|\mathrm{t}] = \begin{bmatrix} 0.88 & 0.22 & -0.41 & -6.72 \\ -0.01 & 0.89 & 0.45 & -11.4 \\ 0.46 & -0.39 & 0.79 & 52.15 \end{bmatrix}$$

$$[\mathrm{R}_{\mathrm{refined}}|\mathrm{t}_{\mathrm{refined}}] = \begin{bmatrix} 0.88 & 0.21 & -0.41 & -6.8 \\ -0.02 & 0.89 & 0.44 & -11.7 \\ 0.46 & -0.38 & 0.8 & 52.3 \end{bmatrix}$$

Table 1 depicts the reprojection error before and after optimization for the four images (B.O.E. = Before Optimization Error and A.O.E. = After Optimization Error)

B.O.E. Mean (pixels)	B.O.*E.* Variance (pixels)	A.O.E. Mean (pixels)	A.O.E. Variance (pixels)
2.23	0.45	1.17	0.49
1.43	0.68	1.26	0.26
1.03	0.28	0.87	0.2
1.68	0.69	1.03	0.43

Table 1: Comparison of errors before and after optimization - Dataset 1

Average Overall error (over all 40 images) before optimization = 1.59 pixels

Average Overall error (over all 40 images) after optimization = 1.07 pixels

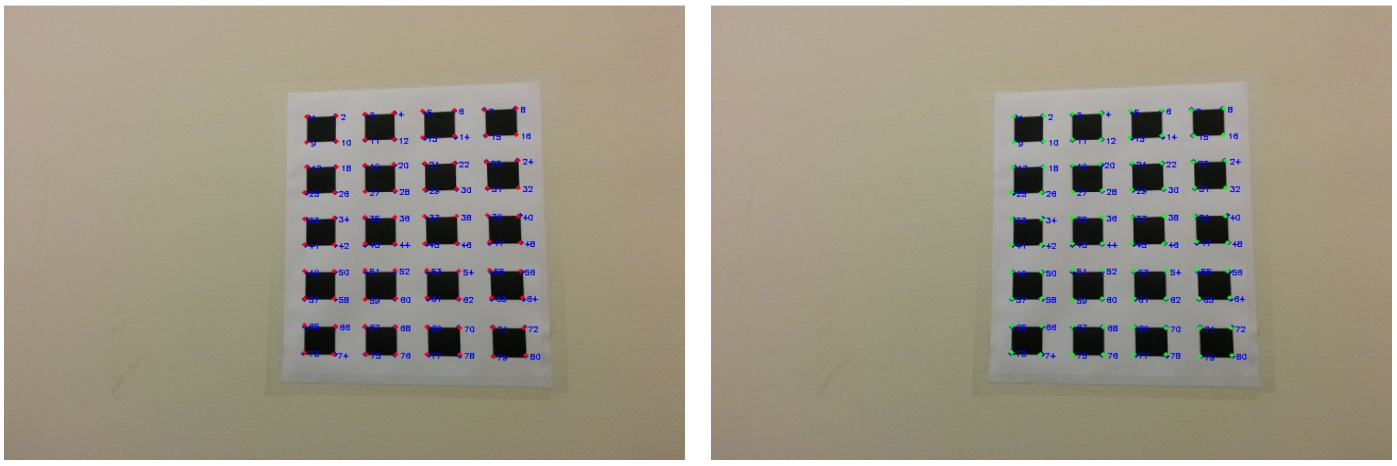
Improvement = 32.7 %

(ii) My Dataset

$$K = \begin{bmatrix} 585.72 & -1.43 & 316.36 \\ 0 & 582.08 & 237.59 \\ 0 & 0 & 1 \end{bmatrix}$$

$$K_{\text{refined}} = \begin{bmatrix} 577.55 & -1.19 & 314.57 \\ 0 & 576.42 & 240.78 \\ 0 & 0 & 1 \end{bmatrix}$$

For the following images the effect of non-linear optimization has been depicted. Blue corners indicate the true location of the corner in the "fixed image", red corners indicate the reprojected corner BEFORE optimization and green corners indicate the reprojected corner AFTER optimization. We can visualize the improvement as the green corners are closer to the blue corner than the red corner.



(a) Before optimization

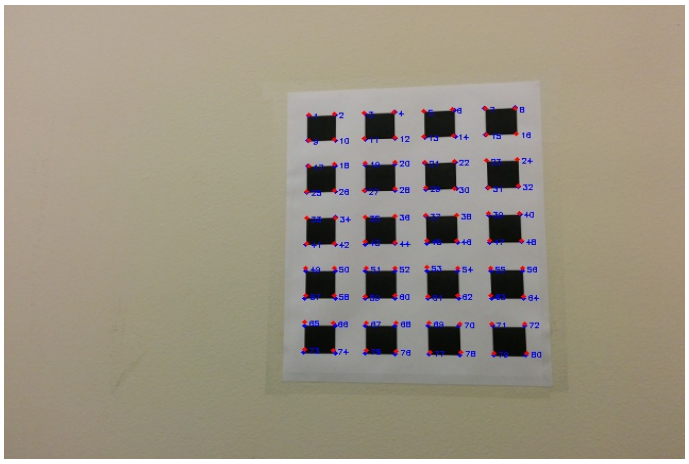
(b) After optimization

Figure 8: Visual verification of optimization (Reprojection of image 3 points)

The extrinsic parameters for this pose of the camera (Image 3 Dataset 2) are:

$$[R|t] = \begin{bmatrix} 0.94 & -0.15 & -0.28 & 3 \\ 0.18 & 0.97 & 0.07 & -15.9 \\ 0.27 & -0.12 & 0.95 & 72.16 \end{bmatrix}$$

$$[R_{\text{refined}}|t_{\text{refined}}] = \begin{bmatrix} 0.94 & -0.15 & -0.29 & 3.22 \\ 0.18 & 0.98 & 0.066 & -16.1 \\ 0.27 & -0.11 & 0.95 & 70.9 \end{bmatrix}$$


(a) Before optimization

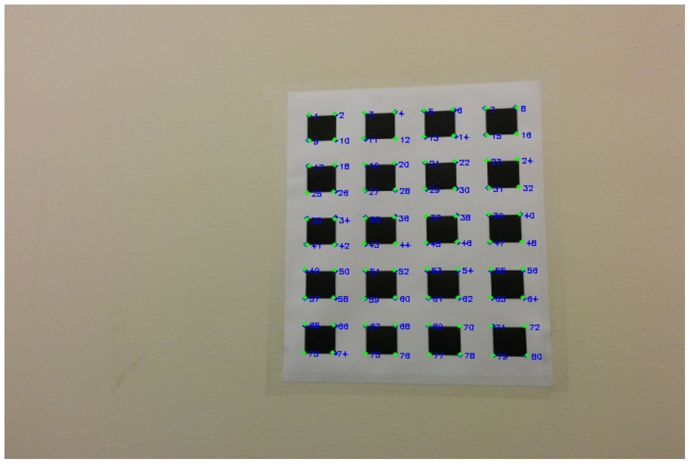
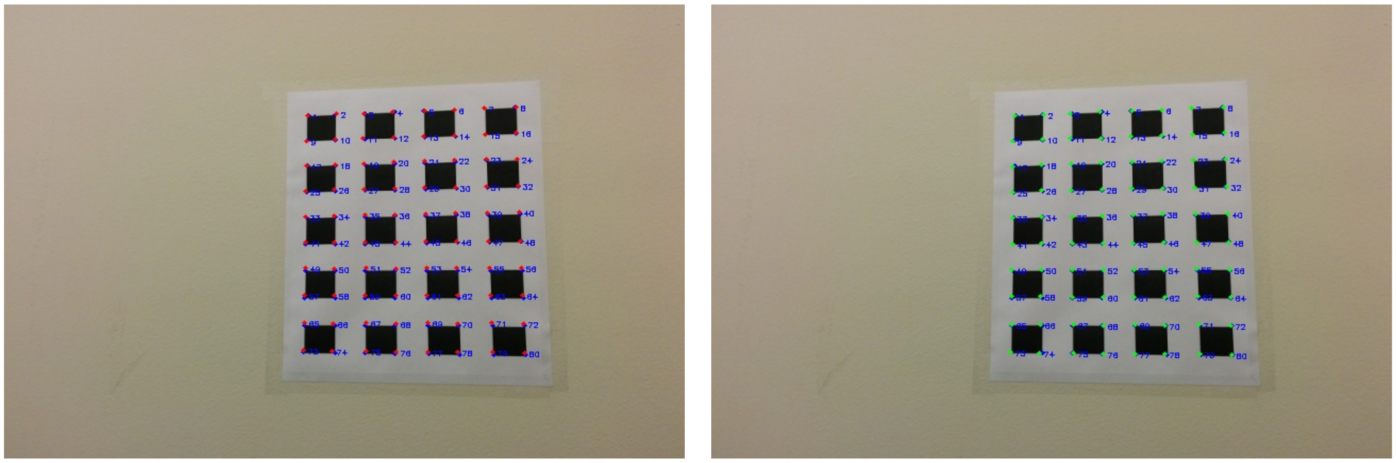

(b) After optimization

Figure 9: Visual verification of optimization (Reprojection of image 6 points)

The extrinsic parameters for this pose of the camera (Image 6 Dataset 2) are:

$$[\mathrm{R}|\mathrm{t}] = \begin{bmatrix} 0.89 & 0.1 & 0.51 & -6.58 \\ -0.21 & 0.96 & 0.16 & -11.02 \\ -0.48 & -0.25 & 0.83 & 61.56 \end{bmatrix}$$

$$[\mathrm{R}_{\mathrm{refined}}|\mathrm{t}_{\mathrm{refined}}] = \begin{bmatrix} 0.85 & 0.1 & 0.5 & -6.45 \\ -0.21 & 0.96 & 0.16 & -11.38 \\ -0.47 & -0.24 & 0.84 & 61.05 \end{bmatrix}$$



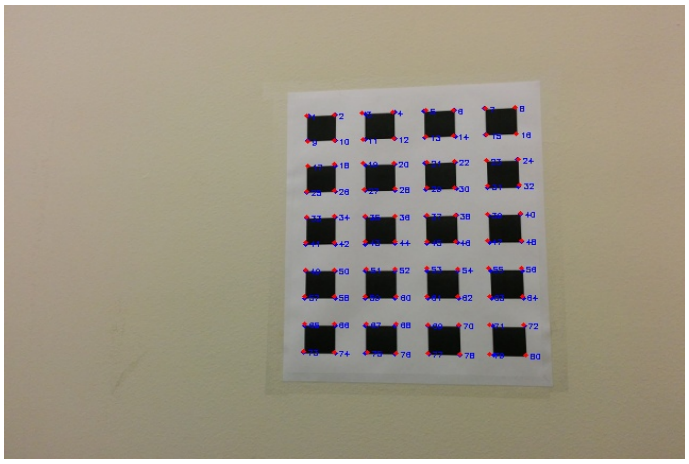
(a) Before optimization (b) After optimization

Figure 10: Visual verification of optimization (Reprojection of image 9 points)

The extrinsic parameters for this pose of the camera (Image 9 Dataset 2) are:

$$[\mathrm{R}|\mathrm{t}] = \begin{bmatrix} 0.93 & -0.014 & 0.35 & -8.29 \\ -0.023 & 0.99 & 0.105 & -9.69 \\ -0.34 & -0.10 & 0.93 & 49.3 \end{bmatrix}$$

$$[\mathrm{R}_{\mathrm{refined}}|\mathrm{t}_{\mathrm{refined}}] = \begin{bmatrix} 0.93 & -0.015 & 0.34 & -8.29 \\ -0.02 & 0.99 & 0.103 & -9.96 \\ -0.34 & -0.1 & 0.93 & 48.82 \end{bmatrix}$$


(a) Before optimization

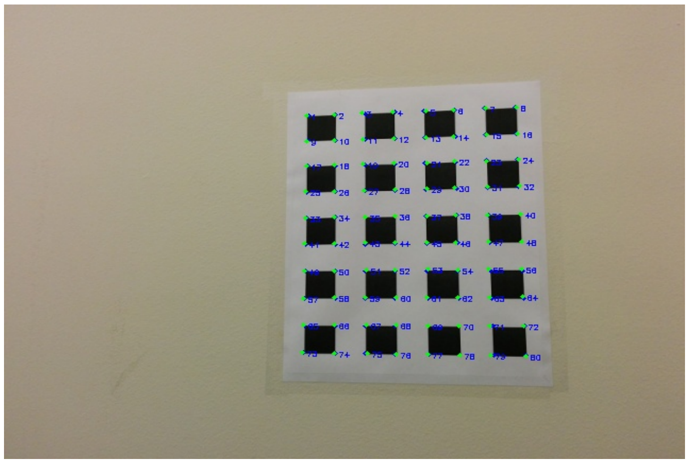

(b) After optimization

Figure 11: Visual verification of optimization (Reprojection of image 12 points)

The extrinsic parameters for this pose of the camera (Image 12 Dataset 2) are:

$$[\mathrm{R}|\mathrm{t}] = \begin{bmatrix} 0.89 & 0.03 & 0.46 & -10.3 \\ -0.11 & 0.98 & 0.14 & -9.83 \\ -0.45 & -0.18 & 0.87 & 47.97 \end{bmatrix}$$

$$[\mathrm{R}_{\mathrm{refined}}|\mathrm{t}_{\mathrm{refined}}] = \begin{bmatrix} 0.88 & 0.029 & 0.46 & -10.2 \\ -0.108 & 0.98 & 0.14 & -10.5 \\ -0.45 & -0.17 & 0.87 & 47.3 \end{bmatrix}$$

Table 2 depicts the reprojection error before and after optimization for the four images (B.O.E. = Before Optimization Error and A.O.E. = After Optimization Error)

B.O.E. Mean (pixels)	B.O.E. Variance (pixels)	A.O.E. Mean (pixels)	A.O.E. Variance (pixels)
1.06	0.22	1.05	0.21
1.65	0.68	0.98	0.23
1.54	0.67	0.92	0.26
1.33	0.49	1.13	0.36

Table 2: Comparison of errors before and after optimization - Dataset 2

Average Overall error (over all 40 images) before optimization = 1.76 pixels

Average Overall error (over all 40 images) after optimization = 1.04 pixels

Improvement = 40.9 %

9. Ground Truth Analysis for My Dataset

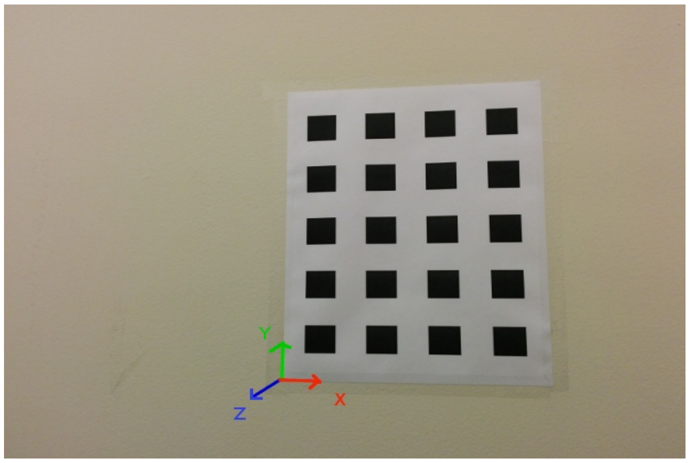


Figure 12: Fixed Image for Dataset-2

$$[\mathrm{R}|\mathrm{t}] = \begin{bmatrix} 0.98 & -0.0015 & 0.14 & -2.86 \\ -0.017 & 0.99 & 0.13 & -10.8 \\ -0.14 & -0.13 & 0.98 & 52.5 \end{bmatrix}$$

$$[\mathrm{R}_{\mathrm{refined}}|\mathrm{t}_{\mathrm{refined}}] = \begin{bmatrix} 0.99 & -0.0012 & 0.14 & -2.72 \\ -0.016 & 0.99 & 0.12 & -11.03 \\ -0.14 & -0.12 & 0.99 & 51.6 \end{bmatrix}.$$

$$\text{Here, R} = \begin{matrix} 0.99 & -0.0012 & 0.14 \\ -0.016 & 0.99 & 0.12 \\ -0.14 & -0.12 & 0.99 \end{matrix} \text{ and t} = \begin{matrix} -2.72 \\ -11.03 \\ 51.6 \end{matrix}$$

What I get along the t makes sense as I was approximately 20.5 inches (52.07 cm) from the pattern. This is along the Z-axis and this is reflected in t_z= 51.6. The t_x and t_y too make sense as I was a little off with respect to the origin in both directions.

10. Effect of conditioning rotation matrix before and during Non-Linear Optimization

I tried two cases: Conditioning BEFORE Optimization and Conditioning DURING Optimization. In both cases, I did not see any change with respect to the mean errors.

Another observation I made was that, when I DID NOT condition the rotation matrix at all, after optimization was performed, my overall error INCREASED by around 10%